Put A1

CLAIMS

1. A method for increasing communication efficiency in a multi-processor system, comprising:

snooping, at a processor having a transition cache and at least one level of cache associated therewith, a first command on a system bus, said system bus providing communication between processors in said multi-processor system, said first command requesting invalidation of a cache line;

generating a second command in response to said first command at one of said levels of cache which stores said cache line if a memory image coherency state for said cache line indicates that said cache line includes modified data, said second command instructing that said cache line be castback;

transferring said second command and said cache line from said one of said levels of cache to said transition cache in response to said first command;

invalidating said cache line in each level of cache associated with said processor that stores said cache line;

snooping a system response to said first command at said processor; and

processing said second command at said processor based on said system response to said first command.

2. The method of claim 1, wherein said processing step discards said second command and said cache line from said transition cache when said system response to said first command is not a retry.

3. The method of claim 2, wherein said processing step converts said second command to a third command in said transition cache if said system response to said first command is a retry, said third command requesting that said cache line be stored in a main memory of said multi-processor system.

Sub A1

4. The method of claim 1, wherein said processing step converts said second command to a third command in said transition cache if said system response to said first command is a retry, said third command requesting that said cache line be stored in a main memory of said multi-processor system.

5. A method for increasing communication efficiency in a multi-processor system, comprising:

generating, at a level one cache in a processor, a first command requiring a reservation;

checking said reservation prior to placing said first command on a system bus, said system bus providing communication between processors in said multi-processor system; and

converting said first command into a second command, which does not require a reservation, when said checking step indicates that said reservation for said first command has been lost.

6. The method of claim 5, further comprising:

transferring said first command from said level one cache to a transition cache in said processor after said generating step;

copying said first command from said transition cache to a system bus controller in said processor; wherein

said checking and converting steps are performed by said system bus controller; and further comprising,

snooping, by said transition cache, said second command on said system bus; and

converting said first command stored in said transition cache into said second command based on said snooping of said second command.

7. The method of claim 5, wherein said first command is an exclusive command and said second command is a non-exclusive command.

8. The method of claim 5, wherein said first command is a store conditional.

put A1

9. A method for increasing communication efficiency in a multi-processor system, comprising:

storing a non-exclusive command associated with a real address in a transition cache of a processor;

snooping, at said processor, a command on a system bus providing communication between processors in said multi-processor system, said snooped command being associated with said real address;

determining, at said transition cache, whether data has started to arrive at said transition cache in response to said non-exclusive command; and

generating a snoop response at said transition cache to said snooped command based on a result of said determining step.

10. The method of claim 9, wherein said generating step does not generate a retry snoop response when said determining step determines that data has not started to arrive at said transition cache in response to said non-exclusive command.

11. The method of claim 10, wherein said generating step generates a retry snoop response when said determining step determines that data has started to arrive at said transition cache in response to said non-exclusive command.

12. The method of claim 10, further comprising:

updating a memory coherency image state for said non-exclusive command at said transition based on said snooped command when said determining step determines that data has not started to arrive at said transition cache in response to said non-exclusive command.

Sub A1

13. The method of claim 9, wherein said generating step generates a retry snoop response when said determining step determines that data has started to arrive at said transition cache in response to said non-exclusive command.

14. A method for increasing communication efficiency in a multi-processor system, comprising:

receiving, at a processor, a first command on a system bus, said system bus providing communication between processors in said multi-processor system, said first command requesting a cache line;

transferring said requested cache line from a cache associated with said processor to a transition cache in said processor as part of a response to said first command;

updating a memory coherency image state associated with said cache line in each cache associated with said processor that stores said cache line;

snooping a system response to said first command on said system bus; and

processing said requested cache line at said processor based on said system response.

15. The method of claim 14, wherein said processing step outputs said requested cache line on said system bus when said system response to said first command is not a retry.

16. The method of claim 15, wherein said processing step converts said response to said first command into a second command for writing said requested cache line in a main memory of said multi-processor system when said system response to said first command is a retry and said memory coherency image state for said requested cache line in said cache which transferred said requested cache line to said transition cache indicated modified data in said requested cache line prior to said updating step.

Sub A1

17. The method of claim 16, wherein said processing step discards said response to said first command when said system response to said first command is a retry and said memory coherency image state for said requested cache line in said cache which transferred said requested cache line to said transition cache does not indicate modified data in said requested cache line prior to said updating step.

18. The method of claim 14, wherein said processing step converts said response to said first command into a second command for writing said requested cache line in a main memory of said multi-processor system when said system response to said first command is a retry and said memory coherency image state for said requested cache line in said cache which transferred said requested cache line to said transition cache indicated modified data in said requested cache line prior to said updating step.

19. The method of claim 18, wherein said processing step discards said response to said first command when said system response to said first command is a retry and said memory coherency image state for said requested cache line in said cache which transferred said requested cache line to said transition cache does not indicate modified data in said requested cache line prior to said updating step.

Sub A1

20. A multi-processor system, comprising:

at least first and second processors;

a system bus providing communication between said first and second processors;

a bus arbiter generating system responses to commands on said system bus; and wherein

said first processor has at least one level of cache associated therewith, a system bus controller controlling communication between said first processor and said system bus, and a transition cache serving as an interface between each level of cache and said system bus controller;

one of said levels of cache associated with said first processor stores a cache line having a memory coherency image state indicating that said cache line includes modified data, and generates a castback command and transfers said castback command and a copy of said cache line to said transition cache when said first processor snoops a first command on said system bus that requests invalidation of said cache line; and

each level of cache associated with said first processor that stores said cache line invalidates said cache line prior to said first processor snooping a system response to said first command.

21. The system of claim 20, wherein said transition cache discards said castback command and said cache line when said system response to said first command is not a retry.

22. The system of claim 21, wherein said transition cache converts said castback command to a second command if said system response to said first command is a retry, said second command requesting that said cache line be stored in a main memory of said multi-processor system.

Sub A1

23. The system of claim 20, wherein said transition cache converts said castback command to a second command if said system response to said first command is a retry, said second command requesting that said cache line be stored in a main memory of said multi-processor system.

24. A multi-processor system, comprising:

at least first and second processors;

a system bus providing communication between said first and second processors;

a bus arbiter generating system responses to commands on said system bus; and wherein

said first processor includes at least a level one cache, a system bus controller controlling communication between said first processor and said system bus, and a transition cache controlling and tracking communication between each level of cache and said system bus controller; and

said system bus controller checks a reservation of a first command, which requires a reservation, generated by said level one cache prior to placing said first command on said system bus, and converts said first command into a second command, which does not require a reservation, when said reservation for said first command has been lost.

25. The system of claim 24, wherein said transition cache receives said first command from said level one cache and communicates said first command to said system bus controller, snoops said second command on said system bus, and converts said first command stored therein into said second command based on said snooping of said second command.

26. The system of claim 24, wherein said first command is an exclusive command and said second command is a non-exclusive command.

27. The system of claim 24, wherein said first command is a store conditional.

Sub A1

28. A multi-processor system, comprising:

at least first and second processors;

a system bus providing communication between said first and second processors;

a bus arbiter generating system responses to commands on said system bus; and wherein

said first processor includes at least one level of cache associated therewith, a system bus controller controlling communication between said first processor and said system bus, and a transition cache controlling and tracking communication between each level of cache and said system bus controller; and

said transition cache determines whether data has started to arrive at said transition cache in response to a non-exclusive command, which is associated with a real address, stored therein when said first processor snoops a command on said system bus which is associated with said real address, and generates a snoop response to said snooped command based on said determination.

29. The system of claim 28, wherein said transition cache does not generate a retry snoop response when data has not started to arrive at said transition cache in response to said non-exclusive command.

30. The system of claim 29, wherein said transition cache generates a retry snoop response when data has started to arrive at said transition cache in response to said non-exclusive command.

31. The system of claim 29, wherein said transition cache updates a memory coherency image state for said non-exclusive command based on said snooped command when data has not started to arrive at said transition cache in response to said non-exclusive command.

Sub A1

32. The system of claim 28, wherein said transition cache generates a retry snoop response when data has started to arrive at said transition cache in response to said non-exclusive command.

33. A multi-processor system, comprising:

at least first and second processors;

a system bus providing communication between said first and second processors;

a bus arbiter generating system responses to commands on said system bus; and wherein

said first processor has at least one cache associated therewith, a system bus controller controlling communication between said first processor and said system bus, and a transition cache controlling and tracking communication between each cache and said system bus controller;

said first processor receives a first command on said system bus requesting a cache line;

one of said caches associated with said first processor that stores said requested cache line copies said requested cache line to said transition cache as part of a response to said first command;

each cache associated with said first processor, that stores said requested cache line, updates a memory coherency image state associated with said requested cache line prior to snooping a system response to said first command; and

said first processor snoops said system response on said system bus to said first command, and processes said requested cache line based on said system response.

34. The system of claim 33, wherein said first processor outputs said requested cache line on said system bus when said system response to said first command is not a retry.

Sub A1

35. The system of claim 34, wherein said first processor converts said response to said first command into a second command for writing said requested cache line in a main memory of said multi-processor system when said system response to said first command is a retry and said memory coherency image state for said requested cache line in said cache which transferred said requested cache line to said transition cache indicated modified data in said requested cache line prior to said updating step.

36. The system of claim 35, wherein said first processor discards said response to said first command when said system response to said first command is a retry and said memory coherency image state for said requested cache line in said cache which transferred said requested cache line to said transition cache does not indicate modified data in said requested cache line prior to said updating step.

37. The system of claim 33, wherein said first processor converts said response to said first command into a second command for writing said requested cache line in a main memory of said multi-processor system when said system response to said first command is a retry and said memory coherency image state for said requested cache line in said cache which transferred said requested cache line to said transition cache indicated modified data in said requested cache line prior to said updating step.

38. The system of claim 37, wherein said first processor discards said response to said first command when said system response to said first command is a retry and said memory coherency image state for said requested cache line in said cache which transferred said requested cache line to said transition cache does not indicate modified data in said requested cache line prior to said updating step.